

CLAIMS

What is claimed is:

1. A method for generating network topology information, the method
5 comprising:

identifying a first link state packet associated with a first node;

identifying a second link state packet associated with a second node;

generating network topology information associated with the first node using
information from the first link state packet; and

10 deferring verification of two-way connectivity checking between the first node
and the second node until analysis of second link state packet is initiated for generating
network topology information associated with first second node.

2. The method of claim 1, wherein the second link state packet is not
analyzed solely for purposes of verifying two-way connectivity.

15 3. The method of claim 1, further comprising:

setting a two-way connectivity flag in response to determining that two-way
connectivity exists between first and second nodes, wherein the two-way connectivity
flag indicates that a valid two-way connection exists between first and second nodes.

4. The method of claim 1, wherein a PATH data structure and a TENT data
20 structure are maintained.

5. The method of claim 1, wherein the first link state packet is scanned
only once.

6. The method of claim 1, wherein the second link state packet is scanned
only once.

25 7. The method of claim 1, wherein the network topology is generated using
Intermediate System-Intermediate System Link State Routing.

8. The method of claim 1, wherein the network topology is generated using
Open Shortest Path First Routing.

9. The method of claim 1, further comprising generating routing table
30 information using the network topology information.

10. The method of claim 4, wherein the PATH and TENT data structures are
rolled-back in response to a determination that two-way link connectivity checking has
failed.

11. The method of claim 1, wherein reverse path insertion is used to generate network topology information in response to a determination that two-way link connectivity checking has failed.

12. The method of claim 4, wherein the PATH data structure is associated with a verification flag used for two-way connectivity checking.

13. The method of claim 4, wherein the PATH data structure is associated with multiple verification flags used for two-way connectivity checking, wherein the number of verification flags is equal to the maximum number of parent nodes.

14. The method of claim 13, wherein a first verification flag is set to true when two-way connectivity is verified between a first node and a second node.

15. The method of claim 14, wherein a second verification flag is set to true when two-way connectivity is verified between a first node and a third node.

16. A method for generating network topology information, the method comprising:

receiving link state packets from a plurality of network nodes;

performing a single scan of the link state packets from the plurality of network nodes to generate network topology information associated with the plurality of network nodes, wherein the single scan performs two-way connectivity checking between two or more network nodes.

17. The method of claim 16, further comprising:

generating routing table information using the generated topology information.

18. A method for generating a network topology, the method comprising:

generating first network topology information using a first link state packet from a first node; and

generating second network topology information using a second link state packet from a second node, wherein generating second network topology information comprises performing two-way connectivity verification between the first node and the second node.

19. The method of claim 18, wherein the second link state packet is scanned only once to generate the second network topology information.

20. A computer program product comprising a machine readable medium on which is provided program instructions for generating a network topology, the computer program product comprising:

computer code for generating first network topology information using a first link state packet from a first node; and

computer code for generating second network topology information using a second link state packet from a second node, wherein generating second network topology information comprises performing two-way connectivity verification between the first node and the second node.

21. The computer program product of claim 20, wherein the second link state packet is not analyzed solely for purposes of verifying two-way connectivity.

22. The computer program product of claim 20, further comprising:
setting a two-way connectivity flag in response to determining that two-way connectivity exists between first and second nodes, wherein the two-way connectivity flag indicates that a valid two-way connection exists between first and second nodes.

23. The computer program product of claim 20, wherein logs of PATH and TENT data structures are maintained.

24. The computer program product of claim 20, wherein the first link state packet is scanned only once.

25. The computer program product of claim 20, wherein the second link state packet is scanned only once.

26. The computer program product of claim 20, wherein the network topology is generated using Intermediate System-Intermediate System Link State Routing.

27. The computer program product of claim 20, wherein the network topology is generated using Open Shortest Path First Routing.

28. The computer program product of claim 20, wherein network topology information is generated using Dijkstra's algorithm.

29. The method of claim 20, further comprising generating routing table information using the network topology information.

30. The method of claim 20, wherein two-way link connectivity checking fails.

31. The method of claim 30, wherein reverse path insertion is used where two-way link connectivity checking fails.

32. An apparatus for generating network topology information, the apparatus comprising:

memory;
at least one processor coupled with memory; and
at least one interface;

the processor being configured to generate first network topology information
5 using a first link state packet from a first node and generate second network topology information using a second link state packet from a second node, wherein generating second network topology information comprises performing two-way connectivity verification between the first node and the second node.

33. The apparatus of claim 32, wherein the processor is further configured to
10 scan a second link state packet once to generate the second network topology information.

34. The apparatus of claim 32, wherein the processor is further configured to set a two-way connectivity flag in response to determining that two-way connectivity exists between first and second nodes, wherein the two-way connectivity flag indicates
15 that a valid two-way connection exists between first and second nodes.

35. The apparatus of claim 32, wherein the processor is further configured to stored into memory a PATH and a TENT data structure.

36. The apparatus of claim 35, wherein the PATH data structure is associated with at least one verification flag used for verifying two-way connectivity
20 checking.

37. The method of claim 35, wherein the PATH data structure is associated with multiple verification flags used for two-way connectivity checking, wherein the number of verification flags is equal to the maximum number of parent nodes.

38. The method of claim 37, wherein a first verification flag is set to true
25 when two way connectivity is verified between a first node and a second node.

39. The method of claim 38, wherein a second verification flag is set to true when two way connectivity is verified between a first node and a third node.

40. The apparatus of claim 32, wherein the processor is configured to scan the second link state packet once.

41. The apparatus of claim 32, wherein the processor is configured to generate network topology using Intermediate System-Intermediate System Link State Routing.
30

42. The apparatus of claim 32, wherein the processor is configured to generate network topology information using Dijkstra's algorithm.

43. The apparatus of claim 32, wherein the processor is configured to generate routing table information using the network topology information.

44. The apparatus of claim 35, wherein the processor is configured to roll-back the PATH and TENT data structures in response to a determination that two-way link connectivity checking has failed.

45. The apparatus of claim 32, wherein the processor is configured to use reverse path insertion to generate network topology information in response to a determination that two-way link connectivity checking has failed.

46. An apparatus for generating network topology information, the apparatus comprising:

means for identifying a first link state packet associated with a first node;

means for identifying a second link state packet associated with a second node;

means for generating network topology information associated with the first node using information from the first link state packet; and

means for deferring verification of two-way connectivity checking between the first node and the second node until analysis of second link state packet is initiated for generating network topology information associated with second node.

47. The apparatus of claim 46, wherein the second link state packet is not analyzed solely for purposes of verifying two-way connectivity.

48. The apparatus of claim 46, further comprising:

means for setting a two-way connectivity flag in response to determining that two-way connectivity exists between first and second nodes, wherein the two-way connectivity flag indicates that a valid two-way connection exists between first and second nodes.

49. The apparatus of claim 46, wherein logs of PATH and TENT data structures are maintained.

50. The apparatus of claim 46, wherein the first link state packet is scanned only once.

51. The apparatus of claim 46, wherein the second link state packet is scanned only once.

52. The apparatus of claim 46, wherein the network topology is generated using Intermediate System-Intermediate System Link State Routing.

53. The apparatus of claim 46, wherein network topology information is generating using Dijkstra's algorithm.

5 54. The method of claim 46, further comprising generating routing table information using the network topology information.

55. The method of claim 46, wherein two-way link connectivity checking fails.

10 56. The method of claim 55, wherein reverse path insertion is used where two-way link connectivity checking fails.

11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000